利用区块链构建公平的安全多方计算 *

黄建华，江亚慧，李忠诚

(华东理工大学 信息科学与工程学院，上海 200237)

摘 要：针对安全多方计算（MPC）中大部分参与者不诚实情况下无法获得公平性这一问题，基于区块链智能合约构造惩罚机制，提出了公平的安全 MPC 协议。协议分为两个阶段，分别为基于可验证秘密共享的 MPC 阶段和公平的秘密重建阶段，参与方只要收集 t+1 个正确份额即可得到最终输出。协议利用同态承诺来验证秘密份额的正确性，使用超时机制来判别恶意参与方的提前终止行为，并对恶意方进行经济惩罚。安全性分析表明诚实参与方能够获得最终输出，否则将得到经济补偿；性能分析表明参与方只需缴纳一轮押金并且大量复杂的秘密份额验证工作都在链下，协议的执行效率得到保证。

关键词：安全多方计算；区块链；智能合约；公平性

中图分类号：TP309.2 doi: 10.19734/j.issn.1001-3695.2018.07.0479

Constructing fair secure multi-party computation based on blockchain

Huang Jianhua, Jiang Yahui, Li Zhongcheng

(*School of Information Science & Engineering*, *East China University of Science & Technology*, *Shanghai* 200237, *China*)

Abstract: This paper proposed a fair secure multi-party computation (MPC) protocol to solve the problem that fairness cannot be achieved when there is no honest majority. The protocol constructed a penalty mechanism based on smart contracts which are stored on the blockchain. It includes the MPC phase based on verifiable secret sharing and the fair secret reconstruction phase. The participants can obtain the final output by collecting just t+1 correct shares. The protocol utilized homomorphic commitments to verify the correctness of the secret shares, employed timeouts to identify the premature abort behaviors of malicious parties, and punished the aborting parties financially. Security analysis shows that honest participants can get the final output, otherwise they will get financial compensation. Performance analysis shows that the protocol requires only one coin-transfer round and a large number of complex secret share verification work is off the chain, which ensures the implementation efficiency.

Key words: secure multi-party computation; blockchain; smart contracts; fairness

0 引言

安全多方计算（secure multi-party computation，MPC）是由图灵奖获得者姚期智提出的百万富翁[1]问题延伸而来，经过 Goldreich、Micali 和 Wigderson 等人[2]的研究发展，安全多方计算已成为国际密码学界研究的热点问题之一。MPC 用于解决一组互不信任的参与方之间保护其隐私的协同计算问题，在安全 MPC 场景中，持有秘密输入的两方或多方，希望共同计算一个函数并得到各自的输出，在这个过程中，除了应得的输出之外，参与方得不到任何额外信息。安全的 MPC 协议除了需要保证输出结果的正确性和参与方输入的隐私性，还要保证公平性，即要么所有人都得到输出要么所有人都得不到输出。

目前研究人员提出了很多高效的 MPC 协议[3~5]，但是这些协议大都只有在应对半诚实攻击者或参与方大多数是诚实者的情况下才能满足上述安全特性，比较明显的例外就是 SPDZ 线的研究成果[6~8]，这类 MPC 协议在大部分参与者都是恶意攻击者的情况下仍能保证协议的输入隐私性和结果正确性。但是，上述高效的 MPC 协议都无法保证公平性[9]。早在 1986 年，文献[10]就已经证明当超过一半的参与者不诚实时，MPC 协议的公平性无法得到保证。目前设计 MPC 协议时考虑得较多的是安全性和正确性，现实世界中协议很难满足公平性，因为恶意参与者会提前终止协议[11]。

安全模型中最常见的就是半诚实模型和恶意敌手模型，这两种模型假设参与者按照初始设定采取行动，半诚实参与者一定会推断其他参与者的秘密信息，恶意参与者一定会无视协议要求破坏协议执行。但是，现实中参与者往往都是理性的，他

收稿日期：2018-07-07；**修回日期：**2018-08-30 **基金项目：**国家自然科学基金面上项目（61472139）

作者简介：黄建华（1963-），男，湖南麻阳人，副教授，博士，主要研究方向为计算机网络与信息安全（jhhuang@ecust.edu.cn）；江亚慧（1994-），女，硕士研究生，主要研究方向为区块链；李忠诚（1994-），男，硕士研究生，主要研究方向为区块链.

们会为自身的利益最大化来采取行动，在适当的激励机制下，所有人（或大多数人）可以不做任何破坏。据此，解决公平性问题的一个新的方法就是在 MPC 协议中加入经济惩罚机制，具体来说就是参与方在执行计算之前先缴纳押金，计算结束后对参与方行为进行验证，诚实者的押金将被退回，恶意者的押金将平分给诚实者。结合惩罚机制来实现公平计算需要解决的一个问题就是如何在没有可信方的情况下管理参与方的押金并对参与方行为进行可信判断。

比特币[12]是第一个去除可信代理的数字加密货币，而以太坊[13] 则通过支持在区块链上编写智能合约实现了对其扩展。区块链是加密货币的底层技术，其通过加密技术和协商共识解决了没有信任基础的参与方之间进行交易的信任问题。针对安全多方计算中大部分参与者不诚实情况下无法获得公平性这一问题[14]，本文提出了基于区块链的公平安全多方计算协议（blockchain-based fair and secure multi-party computation protocol，BFSMPC），该协议使用以太坊创建智能合约，参与方将押金交给智能合约保管，由智能合约判定参与方是否诚实，再对押金做相应的处理，并采用超时机制来判定恶意方提前终止协议的行为。在大多数不诚实参与方的情况下，基于区块链密码学模型的 BFSMPC 可以保证获得公平性。BFSMPC 将大量验证工作放到了链下，进一步简化了智能合约的操作，保证了协议的执行效率。

1 相关工作

MPC 是解决一组互不信任的参与方之间保护隐私的协同计算问题，如何保证安全 MPC 的公平性是目前学术界关注的重点和难点，文献[15]设计了云计算中基于秘密共享的安全 MPC 协议，要求每个服务器得到每个用户的全部盲化输出，之后向全部用户发送输出，这样，用户可以校验来自不同服务器的输出是否一致。但是这种方法并没有真正解决公平性问题，因为在大多数服务器都是恶意的情况下，诚实服务器可能根本得不到用户的盲化输出。文献[10]已经证明了两方计算中无法获得完全公平性，虽然文献[16]的研究成果表明在某些特定的两方计算中完全公平是存在的，但这只是特例，不具通用性，所以有必要考虑对公平性的概念进行弱化。最理想的安全需要满足现实世界和“理想世界”之间的计算上不可区分性，理想世界要能模拟出现实世界的所有攻击行为。理想世界中，因为存在可信的第三方，各参与方都能得到输出。但现实中，公平性只有在大多数参与者都诚实的情况下才能得到保证，文献[17]弱化了这一安全特性，对理想世界进行削弱，使其不再保证公平性，但协议满足计算不可区分性。这类满足除了公平性以外一切安全特性的协议被称为是“安全中止”的。文献[18]的方式则是不改变理想世界，但是放宽了模拟的概念，要求真实世界和理想世界的可区分性最多为 $1/p+negl$，其中 p 是某个规定的多项式，$negl$ 表示一定条件下可忽略的函数。满足这类条件的协议被称为是“$1/p$ 安全的”。文献[19]在标准的真实/理想世界范式下定义了两方安全计算的部分安全性，扩展了公平的密码协议的研究领域。文献[9]提出了可识别的安全中止协议，当恶意方中止协议时，所有参与方都得到通知并识别出谁是恶意中止方。文献[20]提出了 UC 安全的公平安全 MPC 模型，包括了公平的加法理想函数和公平的乘法理想函数，并根据这些模型设计了公平的安全加法协议和公平的安全乘法协议。文献[11,21]借助了原语的辅助来构建公平的安全计算协议，原语担任了可信方的角色，它是通用的，不需要事先知道有关计算的任何信息。其中文献[21]提出的通用黑盒（UBB）原语在两方和多方设置中实现了完备的公平安全计算，它从参与方接收计算电路和一个同意值，对于给定相同电路的参与方集，输出针对该电路的结果，其不足之处在于输入大小和运行时间依赖于目标函数的复杂性。文献[11]表明实现完备公平性不存在“短”的原语，并介绍了公平一致性秘密重建原语，但原语需要多次调用。上述文献都是从协议本身考虑构建公平的安全多方计算，研究结论大多比较消极，与他们不同的是，基于区块链的方案将参与者视为理性的，利用加密货币提供激励机制，促使参与者诚实执行协议。文献[22]利用比特币构造了限时承诺，使得参与者必须在限定的时间内公布其秘密，否则要面临经济惩罚。文献[23]对比特币网络中的属性进行形式化和抽象化，定义了理想函数 F^*_{cr}，在 F^*_{cr}-混合模型中设计出公平的 MPC 协议。文献[24]定义了理想函数 F^*_{ML}，在 F^*_{ML}-混合模型中设计出了常数轮协议。文献[25]对文献[23]作出了改进，降低了脚本的复杂度，减少了链上成本。但是上述协议都是基于比特币网络的，比特币网络不提供图灵完备的语言，无法实现复杂的功能，致使参与方缴纳的押金数达到 $O(N^2)$，对于理想函数的调用数达到 $O(N)$。文献[26]基于以太坊设计智能合约，利用门限秘密共享方案，根据秘密重建情况向黑名单中添加欺骗者，实现 MPC 协议的安全公平进行，押金数为 $O(N)$。文献[27]基于区块链在 UC 模型下构建了具有公平性和鲁棒性的 MPC 协议，但是货币的传输轮数与 MPC 协议轮数相同。

2 公平的安全多方计算协议

BFSMPC 基于以太坊智能合约构建惩罚机制以实现公平性，协议开始时所有参与方需向智能合约缴纳押金，否则协议终止。BFSMPC 主要由两阶段组成，第一阶段中参与方链下执行一个基于 Gennaro 方案[28]的不公平通用 MPC 协议，该协议应用了可验证秘密共享（VSS）方案，秘密被分享在一个 t 阶随机多项式中，最终 n 个参与方得到各自的秘密份额；第二阶段参与方执行一个公平的秘密重建协议，该阶段需要与智能合约进行多轮交互，参与方先链下向其他参与方公布自己的秘密份额，其他参与方验证后，将验证结果反馈给智能合约，由智能合约判定恶意方，最终恶意方押金会被平分给诚实方。该阶段中诚实方只要收集到 t+1 个正确份额即可恢复秘密，如果恢复失败，将得到补偿。

2.1　链下通用 MPC 协议

一般来说，构造通用的安全 MPC 协议主要有三种方法，即基于 Yao 混乱电路的构造方法、基于秘密共享的构造方法和基于同态加密的构造方法[29]。就隐私保护而言，基于秘密共享的通用安全 MPC 协议往往有着更好的性能，可以轻松扩展至云辅助安全计算中，从而使得 MPC 协议更为实用。很多通用的 MPC 协议都是基于半诚实模型，虽然文献[2,9]提出用编译器将半诚实模型下的安全 MPC 协议转换为恶意模型下的安全 MPC 协议，但是在输出阶段，恶意方得到输出后会提前终止协议，诚实方可能得不到最终输出，所以公平性始终无法保障。

为了保证公平性，BFSMPC 在底层通用 MPC 协议输出阶段构建了一个公平的秘密重建协议。文献[25]中使用 n-n 秘密共享方案，承诺方案使用单向散列函数构造，只能保证份额未被窜改，无法保证份额的正确性，这样在秘密恢复的时候，参与者无法分辨其恢复结果是否正确。为解决这一问题，BFSMPC 的底层通用安全 MPC 协议将 Gennaro VSS 方案和 Pedersen 同态承诺方案[30]进行结合，以实现对份额正确性的验证。该通用 MPC 将待计算的函数 $y=f(x_1,x_2,\dots,x_n)$ 表示成由加法和乘法组成的有向图，通过执行相应的加法协议和乘法协议来实现任意的计算，协议大体流程包括三个阶段[31]，如图 1 所示。

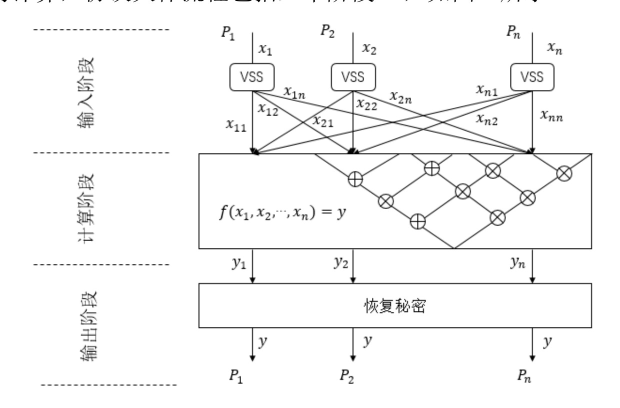


图 1　基于 VSS 的 MPC 协议执行过程

Fig. 1: Execution process of MPC protocol based on VSS

第一阶段为输入阶段，每个参与者使用 VSS 方案在所有参与者中分享自己的秘密输入；第二阶段为计算阶段，参与者执行相应的加法协议和乘法协议，输入都是秘密份额，这些份额均是可验证的，计算阶段输出的是重建 y 所需要的秘密份额，每个协议参与者 P_i 得到份额 y_i；第三阶段为输出阶段，参与者公布自己的份额 y_i，共同将 y 恢复。

2.1.1 输入门

在输入阶段，为了保护隐私，每个参与方 $P_{i(i\in[1,n])}$ 作为分发者（Dealer）利用 Shamir 秘密共享方案将自己的输入分享给其他参与方，并使用 Pedersen 同态承诺方案来保证秘密份额的可验证性。P_i 选择两个随机多项式 $a(x)=\alpha_0+\alpha_1x+\alpha_2x^2+\dots+\alpha_tx^t$ 和 $b(x)=\beta_0+\beta_1x+\beta_2x^2+\dots+\beta_tx^t$，其中 α_0 为秘密值。P_i 将份额 $s_j=(a(j),b(j))$ 秘密地发送给其他参与方 P_j, j=1,2,…,n。同时，P_i 广播承诺值 $A_k=g^{a(k)}h^{b(k)} \bmod p, k=0,1,2\dots n$。其中 p 为 n 个参与方之间协商的一个大素数，满足 $p=2q+1$，q 也是一个素数，g 为 Z_p^* 的 q 阶元，h 为 g 生成的子群中的随机元素。

P_j 收到自己份额后，他需要验证自己的份额是否有效，即验证自己的份额与其他所有人的份额是否在同一个次数为 t 的多项式上（称为 VSPS 性质）。方法如下：

因为 $f(x)=a_0+a_1x+a_2x^2+\dots+a_tx^t$，有

$$\begin{pmatrix}1 & 0 & \cdots & 0\\ 1 & 1 & \cdots & 1\\ \vdots & \vdots & \vdots & \vdots\\ 1 & t & \cdots & t^t\end{pmatrix}\begin{pmatrix}a_0\\ a_1\\ \vdots\\ a_t\end{pmatrix}=\begin{pmatrix}f(0)\\ f(1)\\ \vdots\\ f(t)\end{pmatrix} \tag{1}$$

记为

$$V\begin{pmatrix}a_0\\ a_1\\ \vdots\\ a_t\end{pmatrix}=\begin{pmatrix}f(0)\\ f(1)\\ \vdots\\ f(t)\end{pmatrix} \tag{2}$$

所以

$$\begin{pmatrix}a_0\\ a_1\\ \vdots\\ a_t\end{pmatrix}=V^{-1}\begin{pmatrix}f(0)\\ f(1)\\ \vdots\\ f(t)\end{pmatrix} \tag{3}$$

参与方随机选取 $\delta\in[1,n]$，A_δ 是已公布的承诺，计算

$$A_\delta^{'}=g^{f(\delta)}h^{r(\delta)}=g^{a_0+a_1\delta+a_2\delta^2+\dots+a_t\delta^t}h^{\beta_0+\beta_1\delta+\beta_2\delta^2+\dots+\beta_t\delta^t}=g^{\sum_{j=0}^{t}a_j\delta^j}h^{\sum_{j=0}^{t}\beta_j\delta^j} \tag{4}$$

记 λ_{ji} 为 V^{-1} 的第 j 行第 i 列的值，有 $a_j=\sum_{i=0}^{t}\lambda_{ji}f(i)$，所以

$$A_\delta^{'}=g^{\sum_{j=0}^{t}\sum_{i=0}^{t}\lambda_{ji}f(i)\delta^j}h^{\sum_{j=0}^{t}\sum_{i=0}^{t}\lambda_{ji}r(i)\delta^j}=\prod_{i=0}^{t}(g^{f(i)}h^{r(i)})^{\Delta_i}=\prod_{i=0}^{t}(A_i)^{\Delta_i} \tag{5}$$

其中 $\Delta_i=\sum_{j=0}^{t}\lambda_{ji}\delta^j$。

P_j 将计算得出的 $A_\delta^{'}$ 和之前公布的 A_δ 进行比较，若相等，则检查自己的份额与承诺是否相匹配，若匹配则认为收到的份额是有效的，如果不匹配，则向 Dealer 申请正确的份额。通过这一验证过程，每个参与方都能检查出秘密分发者是否诚实。

安全性分析：在方案中，P_i 广播的承诺中与秘密有关的信息仅为 $A_0=g^{a(0)}h^{b(0)} \bmod p$，对于任意的 $a'(0)\in Z_p$，都存在唯一的 $b'(0)\in Z_p$，使得 $C_0=g^{a'(0)}h^{b'(0)} \bmod p$，$C_0$ 没有泄露关于 $a(0)$ 的任何信息，所以上述过程是信息论安全的。　除此之外，上述方案可以抵抗秘密分发者发送错误的份额给其他参与者这一主动攻击。通过上述推导过程可以看出，如果分发的份额不在同一个 t 次多项式上，是不会通过 VSPS 验证的，因此可以检测出 Dealer 是否诚实。

2.1.2 加法门

假设参与方 $P_{j(j\in[1,n])}$ 拥有通过 VSPS 验证的 α 和 β 的份额 $\alpha_j=f_\alpha(j)$ 和 $\beta_j=f_\beta(j)$ 以及与承诺有关的份额 $\rho_j=r(j)$ 和 $\sigma_j=s(j)$，公开的信息为 $A_0=g^\alpha h^\rho \bmod p$，$A_k=g^{\alpha_k}h^{\rho_k} \bmod p$ 和 $B_0=g^\beta h^\sigma \bmod p$，$B_k=g^{\beta_k}h^{\sigma_k} \bmod p$，其中 k=1,2,…,n。需要计算 $\gamma=\alpha+\beta$，P_j 计算 $\gamma_j=\alpha_j+\beta_j$ 作为自己加法门得到的秘密份额，同时公布对其份额的承诺 $C_j=g^{\gamma_j}h^{\rho_j+\sigma_j} \bmod p$。

安全性分析：因为使用同态承诺方案，$\gamma_j=\alpha_j+\beta_j$ 的承诺为

$C'_j = g^{\alpha_j+\beta_j}h^{\rho_j+\sigma_j} \bmod p = g^{\alpha_j}h^{\rho_j}g^{\beta_j}h^{\sigma_j} \bmod p = A_jB_j$，因为 A_j 和 B_j 已经通过了 VSPS 验证，所以其他参与方都可以对比 C'_j 和 C_j 来验证 C_j 的正确性，判定 P_j 是否诚实，同时所有参与方都可以计算出对 γ 的承诺为 $C_0 = A_0B_0$。除此之外，同输入门中分析一样，此加法协议是信息论安全的，公布的承诺没有泄露关于秘密份额的任何信息。

2.1.3 乘法门

假设参与方 $P_{j(j\in[1,n])}$ 拥有通过 VSPS 验证的 α 和 β 的份额 $\alpha_j = f_\alpha(j)$ 和 $\beta_j = f_\beta(j)$ 以及与承诺有关的份额 $\rho_j = r(j)$ 和 $\sigma_j = s(j)$，公开的信息为承诺 $A_0 = g^\alpha h^\rho \bmod p$，$A_k = g^{\alpha_k}h^{\rho_k} \bmod p$ 和 $B_0 = g^\beta h^\sigma \bmod p$，$B_k = g^{\beta_k}h^{\sigma_k} \bmod p$，其中 k=1,2,…,n。需要计算 $\gamma=\alpha\beta$，方法如下：

$$f_{\alpha\beta}(x) \triangleq f_\alpha(x)f_\beta(x) = r_{2t}x^{2t}+\dots+r_1x+\alpha\beta$$

有

$$V\begin{pmatrix}\alpha\beta\\ r_1\\ \vdots\\ r_{2t}\end{pmatrix}=\begin{pmatrix}f_{\alpha\beta}(1)\\ f_{\alpha\beta}(2)\\ \vdots\\ f_{\alpha\beta}(2t+1)\end{pmatrix} \tag{6}$$

V 是一个非奇异矩阵，记 $\lambda_{j(j\in[1,2t+1])}$ 为 V^{-1} 的第 1 行第 j 列，$\alpha\beta=\lambda_1 f_{\alpha\beta}(1)+\dots+\lambda_{2t+1}f_{\alpha\beta}(2t+1)=\lambda_1 f_\alpha(1)f_\beta(1)+\dots+\lambda_{2t+1}f_\alpha(2t+1)f_\beta(2t+1)$，$P_j$ 计算 $\lambda_j\alpha_j\beta_j$ 并选择两个随机多项式 $h_j(x)$ 和 $u_j(x)$，满足 $h_j(0)=\lambda_j\alpha_j\beta_j$。对于 i=1,2,...,n，P_j 作为 Dealer 将 $(h_j(i),u_j(i))$ 发送给 P_i，并广播承诺 $H(h_j(k)) = g^{h_j(k)}h^{u_j(k)}$，$k$=0,1,…,$n$。

P_j 需要使用零知识证明方法证明其分享的秘密确实是 $\lambda_j\alpha_j\beta_j$，也就是现有公开信息 $A_j = g^{\alpha_j}h^{\rho_j} \bmod p$，$B_j = g^{\beta_j}h^{\sigma_j} \bmod p$ 和 $D_j = g^{\alpha_j\beta_j}h^{\tau} \bmod p$，其中 A_j 和 B_j 是已通过检验的正确承诺，P_j 需要证明 D_j 打开承诺后的值的确是 A_j 和 B_j 打开承诺后的值之积。文献[28]给出了相关的零知识证明方案。

通过零知识证明后保证了正确的秘密份额被分享，然后每一个 P_i 对 P_j 公布的承诺进行 VSPS 验证，通过则检查自己的份额与承诺是否匹配，匹配则认为收到的是有效份额，最终选取 2t+1 个有效份额，计算 $\gamma_i = \sum_{j=1}^{2t+1} h_j(i)$ 作为自己乘法门得到的秘密份额，并公布对 γ_i 的承诺 $H(\gamma_i) = g^{\gamma_i}h^{\theta} \bmod p$。

安全性分析：该乘法方案可以抵抗主动攻击，零知识证明保证了每个参与方 P_j 都正确地产生了 $h_j(x)$，即 $h_j(0)=\lambda_j\alpha_j\beta_j$，之后通过 VSPS 验证 P_j 分发份额的正确性，若 VSPS 验证不通过，说明 P_j 发送了错误份额，其他参与者向 P_j 申请正确份额。对于每个参与方 P_i 在乘法门输出的承诺 $H(\gamma_i)$，其他参与方通过检测其是否等于 $\prod_{j=1}^{2t+1} H(h_j(i))$ 来判定正确性。所有参与方都可以计算出对 γ 的承诺为 $H(\gamma) = \prod_{j=1}^{2t+1} H(h_j(0))$。同输入门中分析一样，此乘法协议是信息论安全的，公布的内容没有泄露关于秘密份额的任何信息。在乘法门中每个参与方需要进行 n-1 次 VSPS 验证，为了提高效率，可以中途不进行 VSPS 验证，而是在所有参与方得到各自乘法门的秘密份额后，再对他们公布的份额承诺进行 1 次 VSPS 验证，但是这种方式是对聚合值进行验证，无法找出具体恶意方，也无法确定恶意方的数量，只能应对被动攻击者，所以如果通过 VSPS 验证，计算继续进行，如果验证不通过，所有参与方在本次乘法门全部重新进行计算并且中途当每个参与方作为 Dealer 分发秘密份额时都要对其进行 VSPS 验证。

2.1.4 输出门

当参与方的秘密输入通过输入门被秘密分享之后，加法门和乘法门都是对这些份额进行运算，不会泄露有关秘密输入的任何信息，且在加法门和乘法门中每个参与方分发份额都要经过 VSPS 验证，这样可以检测出分发者是否诚实，最终输出门的结果是重建 y 所需要的秘密份额，每个参与方 P_i 得到份额 y_i，公开信息为对秘密 y 的承诺 A_0 以及对份额的承诺 A_i，i=1,2,…,n。

通过对输入门、加法门和乘法门的分析可以看出，到达输出门的所有承诺都是经过验证确保是正确的，所以在输出阶段也就是借助区块链实现公平重建阶段，只需要检验参与者公布的份额 y_i 与其之前公开的承诺 A_i 是否匹配，如果匹配，就是正确的份额，只要找到 t+1 个正确的份额，y 就能被正确恢复。

2.2 公平的秘密重建协议

执行完链下通用 MPC 协议后，所有参与方都得到各自的份额，需要公开自己的份额来恢复秘密，但是恶意参与方在得到别人的秘密份额并成功恢复秘密之后，没有公开自己的份额就提前终止协议，可能导致有的参与方得不到最终结果，破坏了 MPC 的公平性。所以需要执行公平的秘密重建协议，检测出恶意参与方并对其进行惩罚，同时对诚实方进行补偿。因为惩罚机制的存在，理性参与者会选择不去作恶。

利用惩罚机制构建公平的 MPC 协议首先要解决押金存储问题。文献[25]提出的公平 MPC 协议基于比特币脚本语言，由于传统比特币区块链的种种限制，没有一个可信方可以存放押金，押金只能通过 claim-or-refund 方式在参与方两两之间传递，总押金数达到 $O(N^2)$，且只能识别出一个恶意方，文献[27]同样基于比特币网络，交易轮数与采用的恶意模型下 MPC 协议的轮数相同。BFSMPC 基于以太坊智能合约实现，协议中有两种账户[32]：外部账户和合约账户，外部账户由用户个人私钥控制，而合约账户由合约代码控制，不为任何人所控制，它可以作为一个可信方来存储押金。在 BFSMPC 中，所有参与方都将押金发送给合约账户，由智能合约进行统一保管，因此缴纳押金轮数只有一轮，总押金数为 O(N)。

BFSMPC 包括本地协议 ConLocal 和智能合约 ConContract。ConContract 由某个参与方 P_i 编写，所有参与方对合约内容达

成一致后，P_i 将其发布到区块链网络，经矿工验证后写入区块链。以下具体描述了 ConLocal 和 ConContract 的内容。ConLocal 由参与方执行，规定了参与方的计算操作以及发送给 ConContract 的数据，ConContract 则由区块链节点执行，负责保管押金及在秘密重建阶段对参与方行为进行判断。因为在区块链网络中，对于智能合约的执行结果需要经过共识，所以为了提高效率，先由参与方本地相互验证各自份额的正确性，如果验证不通过，再将自己的份额交由 ConContract 验证。对于提前终止协议的行为，ConContract 用超时来判定，如果参与方在规定时间内没有向 ConContract 发送信息，则被判为恶意方。

参与方本地协议 ConLocal：

Init：协议参与方集 $P:=\{P_1,P_2,\ldots,P_n\}$，n 个参与方之间协商一个大素数 p，满足 $p=2q+1$，q 也是一个素数，g 为 Z_p^* 的 q 阶元，h 为 g 生成的子群中的随机元素，押金数为$dep，验证数组 J_i 为全 0。

Deposit：确认当前 $T<over(0)$；

将(Deposit, $dep, P_i)发送给 ConContract；

Compute：一旦 $T>=over(0)+2$：

/*为了确保在所有诚实方完成第 ρ 轮任务之前不会有参与方行进到第 ρ+1 轮，任两轮之间至少有两个时钟间隔。*/

向 ConContract 发送(ReadD, P_i)；

接收到 D 后，如果 D!=1，得到押金后协议终止。否则，执行一个不公平的安全 MPC 协议，对于每个 $i\in[n]$，P_i 得到结果 $out_i:=(y_i, r_i, A_0, A_1, \ldots, A_n)$，其中 $A_{j(j\in[n])}:=g^{y_j}h^{r_j}$。

SubCom：一旦 $T>=over(1)+2$：

将(SubCom, P_i, A_i)发送给 ConContract；

SubVerify：一旦 $T>=over(2)+2$：

将(Verify, P_i, y_i,r_i)发送给 $P_{j(j\neq i,j=1,2,\ldots,n)}$；

Verify：一旦接收到来自 $P_{j(j\neq i,j\in[1,n])}$ 的(Verify, y_j,r_j)：

确认 $T<over(3)$，不满足则忽略此消息；

验证 $g^{y_j}h^{r_j}$ 是否等于 A_j，相等置 $J_i[j]$为 1；

验证完所有的份额后，任取 t+1 个正确份额恢复秘密，成功则置 $J_i[0]$为 1；

将(Verify, P_i, J_i)发送给 ConContract；

Check：一旦 $T>=over(4)+2$：

向 ConContract 发送(Check, P_i)；

收到 U 后，如果 $J[0]=1$，向 ConContract 发送(Pay, P_i)，如果押金退还正确，协议终止；如果 $J[0]!=1\&\&J[i]!=1$，向 ConContract 发送(Reverify, P_i, y_i,r_i)；

如果 $J[0]!=1\&\&J[i]=1$ 且恢复秘密成功，一旦 $T>over(5)+2$，向 ConContract 发送(RePay, P_i)；

ReCon：一旦 $T>=over(5)+2$：

向 ConContract 发送(ReturnShare, P_i)；

收到 Sh 后，再次尝试重建秘密；

向 ConContract 发送(RePay, P_i)；

实现公平安全 MPC 的智能合约 ConContract：

Init：恶意方集合 $F:=\varnothing$，恶意方的个数 f 初始化为 0，协议参与方集 $P:=\{P_1,P_2,\ldots,P_n\}$，押金标志位 D:=0，n 个参与方之间协商的一个大素数 p，满足 $p=2q+1$，q 也是一个素数，g 为 Z_p^* 的 q 阶元，h 为 g 生成的子群中的随机元素，押金数为$dep，押金数组 M 记录押金缴纳情况为全 0，状态数组 J 为全 0，数组 Ch 和 Sh 分别用来存储参与方公布的承诺和份额且初始化为空。

Deposit：一旦接收到来自参与方 P_i 的(Deposit, $dep)：

确认当前时间 $T<over(0)$ 且 Ledger[P_i]>= $dep；

确认 P_i 在本次计算中第一次触发 Deposit；

Ledger[P_i]:=Ledger[P_i]-$dep，

Ledger[ConContract]:= Ledger[ConContract]+$dep；

$M[i]:=1$；

如果 $M[1]\&\ M[2]\&\ldots\&\ M[n]=1$，D:=1；

ReadD：一旦 $T>over(0)$：

拒绝接受任何参与方发送的(Deposit, $dep)；

一旦接收到来自参与方 P_i 的(ReadD)：

确认 $T<over(1)$，不满足则忽略该消息；

如果 D!=1，退还押金。

返回 D；

SubCom：一旦接收到 P_i 发送的(SubCom, A_i)：

确认当前时间 $T<over(2)$，不满足则忽略该消息；

记录 P_i 触发了 SubCom；

Ch[i]:= A_i；

一旦 $T>over(2)$：将没有触发 SubCom 的参与方加入 F；

Verify：一旦接收到来自 P_i 的(Verify, J_i)：

确认 $T<over(4)$，不满足则忽略该消息；

记录 P_i 触发了 Verify；

将收到的所有验证数组逐位相与，得到状态数组 $J:=J_1\&J_2\&\ldots\&J_n$；

一旦 $T>over(4)$：将没有触发 Verify 的参与方加入 F；

Pay：如果 $J[0]=1$，对于任意的 $P_i\in P-F$：

Ledger[P_i]:=Ledger[P_i]+ $dep+$(f\cdot \$\mathrm{dep})/(n-f)$)；

Check：$U:=(J,F)$；

返回 U；

ReVerify：一旦接收到来自 P_i 的(Reverify , y_i,r_i)：

确认 $T<over(5)$，不满足则忽略此消息；

如果 $J[0]!=1\&\&J[i]!=1$，验证 $g^{y_i}h^{r_i}$ 是否等于 A_i，相等则 Sh[i]:= (y_i,r_i)，置 $J[i]$为 1；

RePay：一旦 $T>over(5)$：$F:=F\cup\{P_i \mid i\in[1,n]\&\&J[i]!=1\}$；

一旦接收到来自 P_i 的(RePay)，对于任意的 $P_i\in P-F$：

Ledger[P_i]:=Ledger[P_i]+ $dep+$(f\cdot \$\mathrm{dep})/(n-f)$)；

返回 Check；

ReturnShare：返回 Sh；

BFSMPC 执行前先进入预准备阶段，参与方对押金数和其他公共参数达成一致，生成公钥和私钥，广播公共信息（如以

太坊地址和公钥）。假设参与方可以使用秘密通信信道和广播信道，整个协议在同步网络中运行。BFSMPC 的执行流程如图 2 所示。

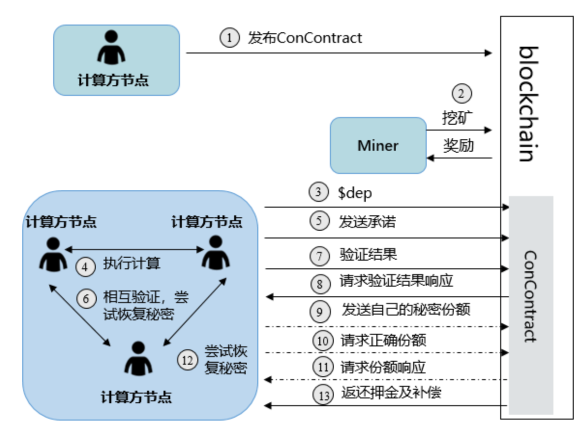


图 2　BFSMPC 执行流程

Fig.2: BFSMPC execution process

a）BFSMPC 一开始进行押金提交，也就是图 4 中的第 3 步。记 $over(\rho)$为第 ρ 轮的结束时间，参与方 P_i 在 $over(0)$之前向 ConContract 发送(Deposit, \$dep, P_i)，ConContract 收到后先检查 P_i 的账户余额是否充足，充足则记录下 P_i 已经缴纳过押金，以后再收到(Deposit, \$dep, P_i)则忽略。当检测到所有人都缴纳押金之后，置标志位 D 为 1。一旦当前时间 $T>over(0)$，ConContract 就不再接受任何(Deposit, \$dep, P_j)，这时如果 D!=1 则将押金退还给所有参与者，协议终止，否则协议继续。

b）参与方 P_i 在 $over(2)$之前向 ConContract 发送(SubCom, P_i, A_i)提交对自己份额的承诺 A_i，ConContract 收到后检测是否满足 $T<over(2)$，满足则将 A_i 存入 Ch[i]中，一旦 $T>over(2)$，将没有发送承诺的参与方加入恶意方集合 F 中。这里对应图 4 的第 5 步。

c）接下来进行链下验证，对应于图 4 中的第 6、7 步。在 $over(3)$之前，参与方 P_i 调用(SubVerify, y_i, r_i)将自己的秘密份额(y_i, r_i)发送给其他所有参与方 P_j（j=1,2,…,n 且 j!=i）。P_j 使用一个验证数组 J_j 来记录自己对其他参与方份额的验证结果，J_j 的初始值为全 0。先检查是否满足 $T<over(3)$，不满足则忽略此信息，否则验证 $g^{y_i}h^{r_i}$ 是否等于 A_i，相等则 $J_j[i]$:=1。P_j 验证完所有的份额之后，如果能取到 t+1 个正确份额来成功恢复秘密，则 $J_j[0]$:=1。P_j 将(Verify, P_j, J_j)发送给 ConContract。

d）ConContract 接受到(Verify, P_j, J_j)后，判断当前时间是否满足 T<over(4)，不满足则忽略此消息，否则记录 P_j 提交了验证数组。ConContract 将收到的所有数组逐位相与，得到状态数组 J。一旦 $T>over(4)$，将没有提交验证数组的参与方加入恶意方集合 F 中。

e）参与方 P_i 向 ConContract 发送(Check, P_i)，收到响应 U=((J,F)后，如果 $J[0]$=1，说明所有参与方都成功恢复了秘密，而不用考虑 $J[i]$的值是否为 1，因为所有参与方都得到了输出，计算成功。此时，若 $P_i \notin F$，F 中恶意方的个数记为 f，P_i 计算自己应得到\$dep+ $(f \cdot \$dep)/(n-f)$ 的退款并向 ConContract 发送(Pay, P_i)，得到退款且确认无误后，协议终止。这里对应图 4 中的第 13 步。

如果 $J[0]$=0，说明有参与方没有成功恢复秘密，这种情况下若 $J[i]$=1，表明 P_i 的份额得到了其他所有参与方的认可，如果 P_i 成功恢复秘密，等到 T>=$over(5)$+2 之后，P_i 向 ConContract 发送(RePay, P_i)；若 $J[i]$=0，说明 P_i 的份额没有得到其他参与方的认可，需要在 $over(5)$之前向 ConContract 发送(Reverify, P_i , y_i, r_i)，这对应于图 4 中的第 9 步。

f）ConContract 接受到(Reverify, P_i , y_i, r_i)后，判断是否满足 $T<over(5)$，不满足则忽略此消息，否则判断若 $J[0]$!=1&&$J[i]$!=1，ConContract 从 Ch[i]中取出 A_i，验证 $g^{y_i}h^{r_i}$ 是否等于 A_i，相等则将(y_i, r_i)存入 Sh[i]，$J[i]$赋值为 1。当 $T>over(5)$时，对于任意的 P_j，如果 $J[j]$!=1，则将 P_j 加入恶意方集合 F 中，这一步检测了两种不诚实行为，第一种为 P_j 提交的秘密份额(y_j, r_j)ConContract 验证不通过；第二种为 P_j 的份额(y_j, r_j)没有被其他参与方认可，P_j 也没有将份额提交给 ConContract 验证。

g）ConContract 接受到(RePay, P_i)后，判断是否满足 $T>over(5)$，不满足则忽略，否则如果检测到 $P_i \notin F$，F 中恶意方的个数为 f，P_i 将收到值为\$dep+ $(f \cdot \$dep)/(n-f)$ 的退款，同时 P_i 会获取到 U=((J,F)，可以验证退款是否正确。这里对应于图 4 中的第 13 步。

h）如果参与方 P_i 没有成功恢复秘密，当 T>=$over(5)$+2 时，P_i 向 ConContract 发送(ReturnShare, P_i)，对应于图 4 中的第 10 步。ConContract 收到后会返回数组 Sh，对应于图 4 中的第 11 步。Sh 保存了链下验证不通过但区块链验证通过的份额，P_i 之前恢复秘密失败说明没有收集到 t+1 个正确份额，其得到 Sh 后可能会凑齐 t+1 个份额，也可能恶意方太多，凑不齐，但在之前几步中已经找出发送错误份额和提前终止协议的恶意方，所以 P_i 会得到补偿，P_i 向 ConContract 发送(RePay, P_i)来请求退款，如果 ConContract 检测到 $P_i \notin F$，F 中恶意方的个数记 f，P_i 将收到值为\$dep+ $(f \cdot \$dep)/(n-f)$ 的退款。协议终止。

3　安全性分析

根据第 2.1 节的推导可以看出 BFSMPC 使用的通用 MPC 协议基于可验证秘密共享方案，可以抵抗主动攻击者。在协议输入阶段，每个参与方作为 Dealer 将秘密份额分发给其他参与方时，会公布对份额以及对秘密的承诺，这个承诺方案是信息论安全的，不会泄露有关秘密的任何信息。除此之外，承诺方案具有同态性，对于 $A_1 = H(\alpha_1, \rho_1), A_2 = H(\alpha_2, \rho_2)$，有 $A_1 A_2 = H(\alpha_1 + \alpha_2, \rho_1 + \rho_2)$ 成立，这样，对于多项式 $f(x)$，当得到对 $f(i)$的承诺，根据同态承诺的性质，可以计算出对多项式系数的承诺，进而可以进行 VSPS 验证，以检测所有份额是否在同一个 t 次多项式上，具体的推导过程见 2.1.1 节。如果验证通过，则可以判定 Dealer 是诚实的，每个参与者都得到了正确的份额。对于乘法门，从 2.1.3 节的推导可以看出每个参与者都需

要作为 Dealer 分发份额，使用零知识证明和 VSPS 验证可以检测出其是否是诚实者。在加法门中，对于 P_i，所有参与方都可以计算出其加法门输出的承诺。链下 MPC 的整个过程中对每个参与方的行为都做了检测，可以识别出恶意参与方，保证 MPC 输出门的承诺是正确的。在公平的秘密重建阶段，只要秘密份额与之前公布的承诺相匹配就认为是正确的份额，收集到 t+1 个正确的份额即可正确恢复秘密，即使有其他的错误份额也不会影响恢复结果，协议具有鲁棒性，BFSMPC 能容忍的最大恶意方数为 n-t-1，因为必须 t+1 个正确份额才能恢复秘密。

为了保证效率，BFSMPC 公平重建阶段先进行链下验证，参与方 P_i 的份额可能不被其他参与方认可，有三种可能，第一种为 P_i 向其他参与方发送了错误份额，第二种为 P_i 发送的是正确份额，但是被其他参与方恶意诬赖，第三种为 P_i 提前退出了协议，没有向其他参与方公布份额。如果链下验证完毕后所有参与方都成功恢复了秘密，这三种恶意行为是不会去检测的，因为已经实现了 MPC 的公平性。如果链下验证结束后，有参与方恢复秘密失败，则任何链下验证不通过的份额都需提交给智能合约 ConContract 验证，ConContract 是自动执行的，不为任何人控制，理性参与方会选择发送正确的份额，因为发送错误份额很容易被检测出来，从而被 ConContract 加入恶意方集，押金就会被没收。对于提前终止协议的行为，区块链使用超时来判断，协议的每一轮都有时限，超出了时限未发送信息则被判为恶意方。对于任一参与者 P_i，ConContract 判定其为恶意方的规则为： a）超时未交纳押金（无法惩罚，协议终止）；b）超时未上传承诺的 A_i；c）超时未上传验证数组 J_i；d）没有通过其他参与方的验证，最终超时未向智能合约提交秘密份额 (y_i,r_i)；e）提交的秘密份额(y_i,r_i)区块链验证不通过。

任何提前终止协议及发送错误信息的参与方都会被 ConContract 识别出来并没收其全部押金，这使得在 BFSMPC 中作恶是有代价的，只要押金数适当，理性的参与方会选择诚实地执行协议。

4 性能分析

基于区块链需要完成共识过程，在区块链上执行操作会耗费大量时间和算力资源，所以应尽可能地将验证操作放到链下。文献[25]中参与者与区块链进行交互的轮数达到 $O(N)$，执行完底层 MPC 协议后，每个参与方得到秘密输出份额，区块链需要验证每个参与方的秘密份额，而且只有每个人都公布正确的秘密份额才能成功恢复秘密。文献[27]中 MPC 协议的每一轮都需要区块链的参与，每一轮参与方都要向区块链证明自己诚实地执行了协议，也就是说每一轮区块链都要验证每个参与方公布的信息，进行大量的零知识证明计算。无论是文献[25]还是[27]，缴纳押金的次数都达到 $O(N)$。

BFSMPC 中第一阶段通用 MPC 协议输出的是通过 VSPS 验证的公开承诺，以及参与方各自掌握的秘密份额。参与方链下验证秘密份额的正确性，因为基于门限秘密共享方案，参与方只需收集 t+1 个正确的份额即可恢复秘密，不需要自己检验的 n 个份额全部正确。区块链只要检测到所有人都恢复了秘密，协议便终止，不需要再检测是否有恶意方链下发送错误份额。只有当有参与方恢复秘密失败时，那些未通过验证的参与方才需要将份额广播给区块链网络，由区块链进行验证，找出恶意方，从而对诚实参与者进行赔偿。因为基于以太坊的 BFSMPC 中存在合约账户，可以用来存储数据和价值，所以只需要协议执行前参与方统一将押金发送到智能合约账户中即可，即 BFSMPC 中参与方只要缴纳一次押金，显著降低了算法的复杂度。

任何基于区块链的解决方案，都要考虑上链的共识时长。在比特币区块链中，产生一个块的时间大约是 10 分钟，为了防止分叉，最终确认要等 6 个块，所以确认一笔交易被写入区块链中需要大概 60 分钟，显然效率太低。文献[25]的协议考虑了确认块的问题，所以每轮时长都非常长。基于以太坊的 BFSMPC 采用 ghost[33]共识机制，15 秒即可产出一个区块，效率比文献[25]的协议更高。

5 结束语

安全 MPC 中，当大多数参与者都不诚实时，公平性无法保证。基于现实中攻击者都是理性的这一特点，采取惩罚措施可以使得恶意攻击者公平地执行协议。区块链具有去中心化，去信任及不可窜改等特点，其上的智能合约不仅可以不为人控制地自动执行代码，存储数据，还能和普通账户一样存储价值。本文利用区块链智能合约构造了惩罚机制，设计了公平的 MPC 协议 BFSMPC。参与方在执行协议前向智能合约缴纳押金，之后，执行一个不公平的安全 MPC 协议，此协议基于 VSS 方案，秘密输出被分享在一个 t 次多项式中，该阶段最终每个参与方得到各自的秘密份额，随后执行一个公平的秘密重建协议，参与者之间互相公布其掌握的份额，并进行链下验证，将验证结果反馈给智能合约，由智能合约判别出恶意方。协议具有鲁棒性，不需要所有份额都正确，诚实方只要收集到 t+1 个正确份额即可恢复秘密，如果秘密恢复失败会得到补偿。安全性分析表明参与方可以通过验证秘密份额的正确性以及与智能合约交互信息的正确性，以得到正确的输出结果；性能分析表明 BFSMPC 只需缴纳一轮押金并且大量复杂的验证操作都在链下进行，相较于文献[25]的算法效率更高。

参考文献：

[1] Yao C. Protocols for secure computations [C]// Proc of the 23rd IEEE Symposium on Foundations of Computer Science. Piscataway, NJ: IEEE Press, 1982: 160-164.

[2] Goldreich O, Micali S, Wigderson A. How to play any mental game [C]// Proc of the 19th Annual ACM Conference on Theory of Computing. New York: ACM Press, 1987: 218-229.

[3] Garg S, Srinivasan A. Two-round multiparty secure computation from

minimal assumptions [C]// Proc of the 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2018: 468-499.

[4] Benhamouda F, Lin H. k-round MPC from k-round OT via garbled interactive circuits [EB/OL]. (2017) [2018-05-03]. https://eprint. iacr. org/2017/1125. pdf.

[5] Cuvelier E, Pereira O. Verifiable multi-party computation with perfectly private audit trail [C]// Proc of the 14th International Conference on Applied Cryptography and Network Security. Berlin: Springer, 2016: 367-385.

[6] Keller M, Pastro V, Rotaru D. Overdrive: Making SPDZ great again [C]// Proc of the 37th Annual International Conference on the Theory and Application of Cryptographic Techniques. Berlin: Springer, 2018: 158-189.

[7] Spini G, Fehr S. Cheater detection in SPDZ multiparty computation [C]// Proc of International Conference on Information Theoretic Security. Berlin: Springer, 2016: 151-176.

[8] Lindell Y, Pinkas B, Smart N P, *et al.* Efficient constant round multi-party computation combining BMR and SPDZ [C]// Proc of the 35th Annual Cryptology Conference. Berlin: Springer, 2015: 319-338.

[9] Ishai Y, Ostrovsky R, Zikas V. Secure multi-party computation with identifiable abort [C]// Proc of the 34th Annual Cryptology Conference on Advance in Cryptology. Berlin: Springer, 2014: 369-386.

[10] Cleve R. Limits on the security of coin flips when half the processors are faulty [C]// Proc of the 18th Annual ACM symposium on Theory of computing. New York: ACM Press, 1986: 364-369.

[11] Gordon D, Ishai Y, Moran T, *et al.* On complete primitives for fairness [C]// Proc of the 7th Theory of Cryptography Conference. Berlin: Springer, 2010: 91-108.

[12] Nakamoto S. Bitcoin: a peer-to-peer electronic cash system [EB/OL]. (2008) [2018-05-03]. http://bitcoin. org/bitcoin. pdf.

[13] Singhal B, Dhameja G, Panda P S. Beginning blockchain [M]. Berkeley: Apress, 2018: 219-266.

[14] Kosba A, Miller A, Shi E, *et al.* Hawk: the blockchain model of cryptography and privacy-preserving smart contracts [C]// Proc of IEEE Symposium on Security and Privacy. Piscataway, NJ: IEEE Press, 2016: 839-858.

[15] Jakobsen T P, Nielsen J B, Orlandi C. A framework for outsourcing of secure computation [C]// Proc of the 6th Edition of the ACM Workshop on Cloud Computing Security. New York: ACM Press, 2014: 81-92.

[16] Gordon S D, Hazay C, Katz J, *et al.* Complete fairness in secure two-party computation [J]. Journal of the ACM, 2011, 58 (6): 1-24.

[17] Goldreich O. Foundations of cryptography: volume 2, basic applications [M]. Cambridge: Cambridge University Press, 2004.

[18] Katz J. On achieving the best of both worlds in secure multiparty computation [C]// Proc of the 39th Annual ACM Symposium on Theory of Computing. New York: ACM Press, 2007: 11-20.

[19] Gordon S D, Katz J. Partial fairness in secure two-party computation [C]// Proc of the 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2010: 157-176.

[20] 田有亮，彭长根，马建峰，等. 通用可组合公平安全多方计算协议 [J]. 通信学报, 2014 (2): 54-62. (Tian Youliang, Peng Changgen, Ma Jianfeng, *et al.* Universally composable secure multiparty computation protocol with fairness [J]. Journal on Communications, 2014 (2): 54-62.)

[21] Fitzi M, Garay J A, Maurer U M, *et al.* Minimal complete primitives for secure multi-party computation [J]. Journal of Cryptology, 2005, 18 (1): 37-61.

[22] Andrychowicz M, Dziembowski S, Malinowski D, *et al.* Secure multiparty computations on bitcoin [C]// Proc of IEEE Symposium on Security and Privacy. Piscataway, NJ: IEEE Press, 2014: 76-84.

[23] Bentov I, Kumaresan R. How to use bitcoin to design fair protocols [C]// Proc of the 34th Annual Cryptology Conference on Advance in Cryptology. Berlin: Springer, 2014: 421-439.

[24] Kumaresan R, Bentov I. How to use bitcoin to incentivize correct computations [C]// Proc of the 21st ACM Conference on Computer and Communications Security. New York: ACM Press, 2014: 30-41.

[25] Kumaresan R, Vaikuntanathan V, Vasudevan P N. Improvements to secure computation with penalties [C]// Proc of the 23rd ACM Conference on Computer and Communications Security. New York: ACM Press, 2016: 406-417.

[26] Zyskind G. Efficient secure computation enabled by blockchain technology [D]. Cambridge: Massachusetts Institute of Technology, 2016.

[27] Kiayias A, Zhou H S, Zikas V. Fair and robust multi-party computation using a global transaction ledger [C]// Proc of the 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2016: 705-734

[28] Gennaro R, Rabin M O, Rabin T. Simplified VSS and fast-track multiparty computations with applications to threshold cryptography [C]// Proc of the 17th Annual ACM Symposium on Principles of Distributed Computing. New York: ACM Press, 1998: 101-111.

[29] 蒋瀚，徐秋亮. 基于云计算服务的安全多方计算 [J]. 计算机研究与发展, 2016, 53 (10): 2152-2162. (Jiang Han, Xu Qiuliang. Secure multiparty computation in cloud computing [J]. Journal of Computer Research and Development, 2016, 53 (10): 2152-2162.)

[30] Pedersen T P. Non-interactive and information-theoretic secure verifiable secret sharing [C]// Proc of Annual International Cryptology Conference on Advances in Cryptology. Berlin: Springer, 1991: 129-140.

[31] 邱卫东，黄征. 密码协议基础 [M]. 北京：高等教育出版社, 2009: 156. (Qiu Weidong, Huang Zheng. Cryptographic protocols [M]. Bei Jing: Higher Education Press, 2009: 156.)

[32] Jameson H. Accounts, transactions, gas and block gas limits in ethereum [EB/OL]. (2017) [2018-05-21]. https://hudsonjameson. com/2017-06-27-accounts-transactions-gas-ethereum/.

[33] Sompolinsky Y, Zohar A. Secure high-rate transaction processing in bitcoin [J]. Computer Science, 2015, 8975: 507-527.